MODIS QUARTERLY REPORT
JULY-SEPTEMBER 1992

DR. ROBERT H. EVANS
UNIVERSITY OF MIAMI
RSMAS/MPO

Contract Number: NAS5-31362

==

Due to the interlocking nature of a number of projects, this and subsequent reports will contain coding to reflect the funding source. Modis funded activities are designated with an M, SeaWiFS with an S, Pathfinder with a P, and Headquarters with an H. There are several major sections within this report; Database, client/server, matchup database, and DSP support.

A. NEAR TERM OBJECTIVES
B. OVERVIEW OF CURRENT PROGRESS
C. FUTURE ACTIVITIES
D. PROBLEMS

A. NEAR TERM OBJECTIVES

A.1 Modis Objectives (M)

A.1.1. Continue to develop and expand the processing environment
- a. increase computational efficiency through concurrent operations
- b. determine and apply more efficient methods of data availability for processes

A.1.2. Begin extensive testing using global CZCS and AVHRR GAC data with database processing to test the following:
- a. algorithm capability
- b. machine and operating system stability
- c. functionality required for the processing and analysis environment

A.2 SeaWiFS Objectives (S)

A.2.1. Continue testing of processing methodology.
A.2.2. Continue to develop relationship between database and in-situ environment.

A.3 Pathfinder Objectives (P)

A.3.1. Expand matchup database as applicable.
A.3.2. Continue testing of methodology.

A.4 DSP Objectives (H)

A.4.1. Continue testing of processing methodology.
A.4.2. Continue to expand the number of sites supported.
A.4.3. Expand the supported hardware/software platforms

B. OVERVIEW OF CURRENT PROGRESS

B.1 Automatic Processing Database (S)

The record–addition subroutine (add main per) was modified to work through the client/server. This program transfers a large amount of character data. To avoid the somewhat inefficient and difficult-to-modify transition from FORTRAN to a C data structure in the server, then back to FORTRAN for the database interface, the client side was modified to create an ASCII file with the data to be transferred. The name of this file is passed through the server, the interface then reads the file and adds the records to the database.

A new process initiation program was written (addrecdir); it functions as follows. It is a continuously running daemon that periodically checks a staging directory. When files appear in that directory, addrecdir spawns a job to run the program that adds the MAIN and PROCESS_CONTROL records to the database. After records have been added, it then renames the file into a run directory for processing. When all files in the staging directory have been processed, it sleeps for a specified time, then periodically rechecks the staging directory.

Code from the CZCS ingester used to add records to the database was moved into a new ingester, PATHTIROS, used for AVHRR GAC data. The GAC passes will be split into pieces for processing, and each piece is tracked separately, with what is called a child record. Prior to modification, the program created a single child record each time the ingester was called, controlled by an input file. PATHTIROS has been modified to break up the pass into segments containing a fixed number of scan lines (tracked in the database by BEGSCAN and ENDSCAN); the code used to create the parent/child records has been tested and extended. Parameters are read from the environment to control the size of pieces and any overlap, and all child records are added at one time. When each piece of a pass (child) is processed, the database is notified. When all child records have been processed correctly, a mosaic job is triggered to put the pieces back together. This was modified to trigger multiple jobs (in this case, to create a day? night, and day/night mosiac). The ability to trigger a third level was added, in this case, a ‘CLEAN’ function that removes intermediate files after the three mosiac jobs have completed.

The PROCEDURE, PROCESS STEPS and PARAMETERS tables in the database were modified to accommodate the new PATHTIROS processing scheme and command files. The ingester breaks up the pass into pieces defined by an input control file. Previously, a small number of these input files were used to break up GAC passes. The PATHTIROS procedure was modified to use two variables from the database, BEGSCAN and ENDSCAN, to create an input file on the fly.

Once all the pieces of the system were assembled, testing began and problem areas identified. The most frequent failures were due to momentary drop-out of NFS-mounted disks. This drop-out can affect many parts of the processing, depending on what is being executed when a disk disappears. Many files are written and read during a processing job, and programs and procedures have been (and are still being) modified to handle these situations. Most of these cases are being handled by a retry loop. When an error occurs in creating or opening a file, an 'err=' option loops to try the operation again, to a maximum of five tries. The trouble spots identified so far are: add_main_pcr: creation and opening of the data transfer file; db Request: creation of the DSP procedure file (*.dsp) and the UNIX shell script (*.sh) that runs it.

The drop-out problem can also occur during the processing itself, and steps are being taken to trap these errors. Occasionally, DSP itself does not initialize properly. (It has not been determined for certain that this is due to disk drop-out.) The shell script that db Request writes has been modified to check for proper initialization, and to retry once. The db Report step has been moved into the shell script, so a report will be made if the *.dsp file itself is not found. The error variable (status) and error message (AUTOPROC ERROR) are set to indicate an error before calling the *.dsp procedure file, directing an appropriate message to the database. In the *.dsp procedure file itself, the status is set to failure and the error message identifies the procedure to be called before any subcommand file is called. The subcommand file resets the status to good if it completes successfully. The status is checked when control returns to the *.dsp procedure file. If the status is good, processing continues. If the status shows a failure, control returns to the shell script, which reports the appropriate error status and message. If the entire process completes successfully, the status is set to good, the error message is blanked, and a success report is returned to the database.

B.2 Client/Server Status (S)

The initial processing scheme was based on a sequential flow of processes related to a single satellite pass. Early efforts yielded one day's data processed per day; this was improved slightly to two day's data per day. As a result of analyzing multiple tests, which are described below, several factors were recognized as contributing to pace of processing. The normal processing sequence involves the acquisition of the data, storage of the files on an optical jukebox, database population, SST processing, global binning, and finally cleanup. The sequence is governed by data, processing resource, and storage availability. These factors are examined further below. However, the result of applying the lessons learned has been an increase in overall processing by concurrently processing multiple orbits to the point that we a processing 6 day's data per day and have set a goal of 10 day's data per day.

During August and September period, the following tasks were completed:

1. Test run of ten day's worth of data in a sequential manner.

2. System modification to facilitate concurrent processing.

3. Additional testing of concurrency improvements after modification.

In August and September, our basic goal was to continuously test, to determine problem sources, and to tune and optimize the system for efficiency and speed.

The first test we ran was successful using six master control processes (mcp's) running together. This test did not, however, make full use of the concurrent processing potential available within the system. There were three reasons why the potential was not used.

The first was that the previous mcp system design allowed data files to be loaded into the database when there were no unfinished jobs in the database. This forced data files to be processed sequentially one at a time. With multiple mcp's running concurrently, there was a waste of the processing resources during each of the processing cycles.

The second reason was the design of data flow requiring satellite information to be processed in three stages. A satellite pass is divided into a number of pieces; these pieces are processed before the second stage that performs both the postage stamp and time binning functions. The third step is cleanup job for each path that must wait for the completion of prior processing steps.

The third reason was the location of the data files. These files resided in a directory that dsp command procedures had to access through NFS. This meant process was limited by 1/0 speed and was capped by the NFS transfer rate.

After examining system performance and determining the causes listed above, we modified our mcp and the database supporting programs so that more concurrency could be achieved.

First, we removed the mechanism in the system requiring only one data file at a time and added new processing capability. This new program checks the designated directories to determine whether new data files need to be moved into place based on the following conditions:

1. Are input data files are available from jukebox ? i.e. Is new data available for processing.

2. Are there are fewer files in the designated directories than specified? To achieve a greater degree of concurrency, data must be available in the database for processing; with data available, the mcp's will spend less time idle. The restriction is based on available disk space; each input file requires more than 40 M bytes. By scaling the available free space using the 40 M byte

figure, a determination can be made concerning the number of files that can be moved into the directory. By increasing the available space and properly managing the resources, the throughput bottleneck caused by idle mcp's will be greatly reduced.

Second, the allocation of tasks among six mcp's was modified. As mentioned above, the processing has been divided into a number of discrete steps: first the processing the pieces of input data from an orbit, second the space (postage stamp) and time binning, and third the cleanup. During the early tests, we realized that a bottleneck was occurring due to a lack of resources during the binning steps. As a result, we modified the system such that two of the mcp's will be designated to do the binning.

In addition, to avoid the bottlenecks associated with the NSF transfers, data files are being transferred concurrently, via ftp, to the disk local to the processing CPU.

The cleanup steps for each path have also been modified. If the directories are not purged of accumulated files after each path, we will quickly run out of space. In the previous system, the clean job had been designed to operate on a single processing stream; when applied to the concurrent scheme, it did not function efficiently or thoroughly. It either deleted all the files with a designated suffix in a directory or did not delete them at all. For concurrent processing, the cleanup job needed a list of the file names to be removed. This capability has been added to the system and the cleanup job is functioning properly.

After these modifications, we ran additional tests and obtained much better results. We have achieved an increase in concurrency and obtained some insight into avenues for further tuning the system.

B.3 Matchup Database (P)

The derivation and validation of improved sea surface temperature (SST) algorithms require a matchup database (MDB) of AVHRR and in situ measurements coincident (or nearly coincident) in space and time. The first step towards the development of matchup databases was the construction of what we termed an RexperimentalS matchup database. The experimental MDB included in situ SST observations from moored buoys off the east coast of the United States and in the Gulf of Mexico. The satellite data were extracted from high-resolution AVHRR imagery archived at the University of Miami. Details on the construction of the experimental MDB were provided in previous reports.

The development of global SST algorithms for the Pathfinder project requires that a variety of environmental conditions be considered, from tropical to subpolar regimes. This requires that an operational global matchup database (GMDB) be compiled with a wide geographic distribution of in situ SSTs. Our efforts during the past quarter were concentrated on the construction of a global MDB for the test data year, 1988.

The construction of the global MDB relied on the tools and techniques developed for the experimental MDB. At the same time, new software had to be developed. For instance, the satellite data extractions for the experimental MDB were performed using high-resolution direct broadcast data. To do extractions using the GAC files, new procedures had to be developed and tested. The following paragraphs summarize the main activities conducted during this quarter in relation with the matchup databases.

1. In situ SST data. The various sources of in situ SST observations have been described on previous reports. The global in situ SST records were processed through the TCAP (Time of Closest Approach) filter previously described. Briefly, the purpose of the filter is to identify which in situ locations/times were coincident (or nearly coincident) with AVHRR observations. About 100,000 SST records from various sources (moored buoys and drifters) passed the filter and were used to extract the corresponding AVHRR/GAC data. Figures B.3.1 through B.3.5 show the geographic distribution of all the 1988 in situ SST records that passed the TCAP filter. It is apparent that temperate and tropical regimes are likely to be well represented in the global MDB. Subpolar conditions are not as common, as only a few drifting buoys are found in this regime.

In this period we also investigated an apparent problem with in situ SST reports from drifting buoys. The problem was in the data set compiled by the Canadian Marine Environmental Data Service (MEDS). The times and locations for some of the MEDS drifters were not congruent. For instance, a drifter in the southeastern Indian Ocean reported a position at time X. Time X was about 27 minutes after than the TCAP prediction. An orbit plot revealed that at time X the NOAA spacecraft was flying over the southeastern Pacific, where it could not have received a transmission from the buoy. Because the anomalous records represent a small portion of the available in situ SST data, and the source of errors could not be identified, the safest course of action is to eliminate them.

We also proceeded with the compilation and reformatting of in situ data for other Pathfinder years. Additional drifting buoy data for 1984-1987 were obtained from Dr. C. McClain (NASA/GSFC) and merged with the previous datasets.

2. AVHRR/GAC data. During this quarter we performed, for the first time, extractions of AVHRR data at the in situ times/locations that passed the TCAP filter. Test extractions were performed for a few days in January 1988. Close examination of results from the test extractions revealed problems in the extraction process, which were subsequently corrected. A second round of test extractions was performed for a longer period (10 days) , and results were satisfactory. While checking the performance of the extraction procedures, we encountered some missing GAC orbits for

some of the test days; we need to investigate whether this is a frequent occurrence. A final test extraction for an even longer period (one month) was performed for timing purposes. Results from that extraction are now being analyzed. The tests suggest that the GAC extractions will require slightly less than a week per data month. That is, to extract, data for the whole test year would require 2-3 months. Before the operational extractions proceed, an automated correction for drift in the spacecraft clock needs to be added, in order to improve the geolocation of the data.

3. Matchup procedures. The number of matchup points in the global MDB is going to be much higher than in the experimental MDB. We anticipate that about half of the 100,000 points for which extractions will be performed will be usable (most of the missing half will correspond to cloudy pixels) . We are currently developing procedures for subsampling the matchup database, in order to reserve a portion of the matchups solely for algorithm validation purposes. As a first step, procedures were developed to allow the identification of the type of in situ platform, and the data source. This will allow us to ensure that the proportion of data of each type is homogeneous between estimation and validation matchup databases.

B.4 DSP Support (H)

The following section provides summary of DSP support activities for the third quarter of 1992.

B.4.1 Testing:

More testing of the PATH programs (PATHSST, PATHBIN, PATHTIME, PATHMOS). Using client/server/rdb database to process GAC data on SGI. Updating test command procedures to test DSP programs on all five supported machines. B.4.2 Modifications/Additions to DSP:

Started making changes to support more than four image planes. Started making changes to support up to 4096 pixels per line. Added landscape option to MIA2PS. Added a new function, Dsp_IsPlane, so programs can find out if an input or Output image is a file on disk or an image plane. Converted sealwio.dsp to UNIX. Made a temporary 9 km binner and mosaic for CZCS data. B.4.3 Problems fixed:

Misc. fixes to libraries.
Misc. fixes to GETCOM (e.g. clean up dead DSP'S) Misc. fixes to XFBD Fixed DSP parsing of filenames without extensions. Fixed handling of menus and help pages. Fixed some problems with communications between programs. Fixed double-quote handling in operating system commands. Fixed handling of the navigation height associated data. Fixed F for case where consecutive passes are 6.5 minutes apart; increased maximum number of records per day; and fixed some error checking. Fixed SAR command procedure to use the proper input when only remapping. Fixed default input image name for SHRINK. Fixed PATHTIME to handle different quality data

properly Fixed GSFCBIN to not create an output file if there is no data to bin.

B.5 Team Interactions

C. FUTURE ACTIVITIES

C.1 Database Future Work

The GAC and CZCS processing will continue to be run to identify further trouble areas. The error traps will be refined and new ones added where needed.

Now that the major programs are essentially complete, a number of minor functions will be added to the client/server process. Programs used before to reset the database after individual error or computer crashes will be adapted, and new ones written as needed. Such functions as priority changes, monitoring, establishing new links would be useful. A template client is begin developed that will facilitate the addition of functions without interfering with established clients.

The current method of creating procedures, process steps and parameters is cumbersome and, at times, confusing. An easier,way to define new processing threads, perhaps using a GUI (Graphical User Interface) will be developed.

The ability to trigger one class of jobs upon completion of another class of jobs has shown itself to be useful. Currently, however, there are only two types of triggers: a parent/child class triggering a mosaic class, and a mosaic class triggering a cleanup. It is possible to modify the PROCEDURE table to be able to define automatic triggering of follow-on jobs in a more flexible manner.

Two major tasks still remain: the interface upgrade and documentation.

The changes to the FORTRAN interface were described in the June MODIS report, and will not be repeated here.

The documentation process is in the barest of rudimentary phases. When the modifications began on the interface, there was no documentation except for comments in the source, which primarily referred to variable definition.

In the intervening months, the purpose and use of the subroutines has been determined, but not fully documented. In addition, the changes and enhancements have only been ‘documented’ in reports such as this. No coherent, complete document exists which explains how to use the autoprocessing system, or how it works. So, even though the system is not yet in its final form, the process of documentation should be started. There should be essentially two classes of documentation: one for uses, who want to run the system, create processing threads and modify existing procedures' paths, and one for programmers who want to make changes to or extensions of the code itself. Such things as an installation manual, bug-reporting procedures and addition of error messages will need to be developed.

C.2 Client/Server Future Work

C.2.1. Creation of a resource manager and a performance monitor.

C.2.2. Expansion of the error handler to provide broader coverage and to integrate into the overall system error recovery scheme.

C.2.3. Continue testing the client/server with CZCS and AVHRR data. This would include the acquisition of a UNIX resident database to run parallel tests.

C.2.4 Continue enhancement of processing efficiency through greater use of concurrent processing.

C.2.5 Examine other processing schemes to determine which elements could be either included or adapted for use within the client/server concept.

C.3 Pathfinder (P)

C.3.1. Continue development of linking processes between in-situ and processed satellite data.

C.3.2. Expand the validation dataset.

C.4 Headquarters (H)

C.4.1. Create tools to assist in results interpolation.

C.4.2. DSP – Fix programs that access the graphics plane to use the navigation from the input image and not the graphics plane.

C.4.3 Refine PATH binning and mosaic pixel quality algorithm to eliminate clouds.

C.4.4 Verify workstation DSP (SGI, SUN, DECstation, VAXstation) by comparing each program's output with the Adage system.

C.5 Modis (M)

C.5.1. Continue working with H. Gordon on an implementation of prototype ocean color atmospheric correction algorithms.

C.5.2 Continue working with D. Clark on in-situ database requirements .

D. PROBLEMS

D.1 Database Problems

None listed separately

D.2 Client/Server Problems

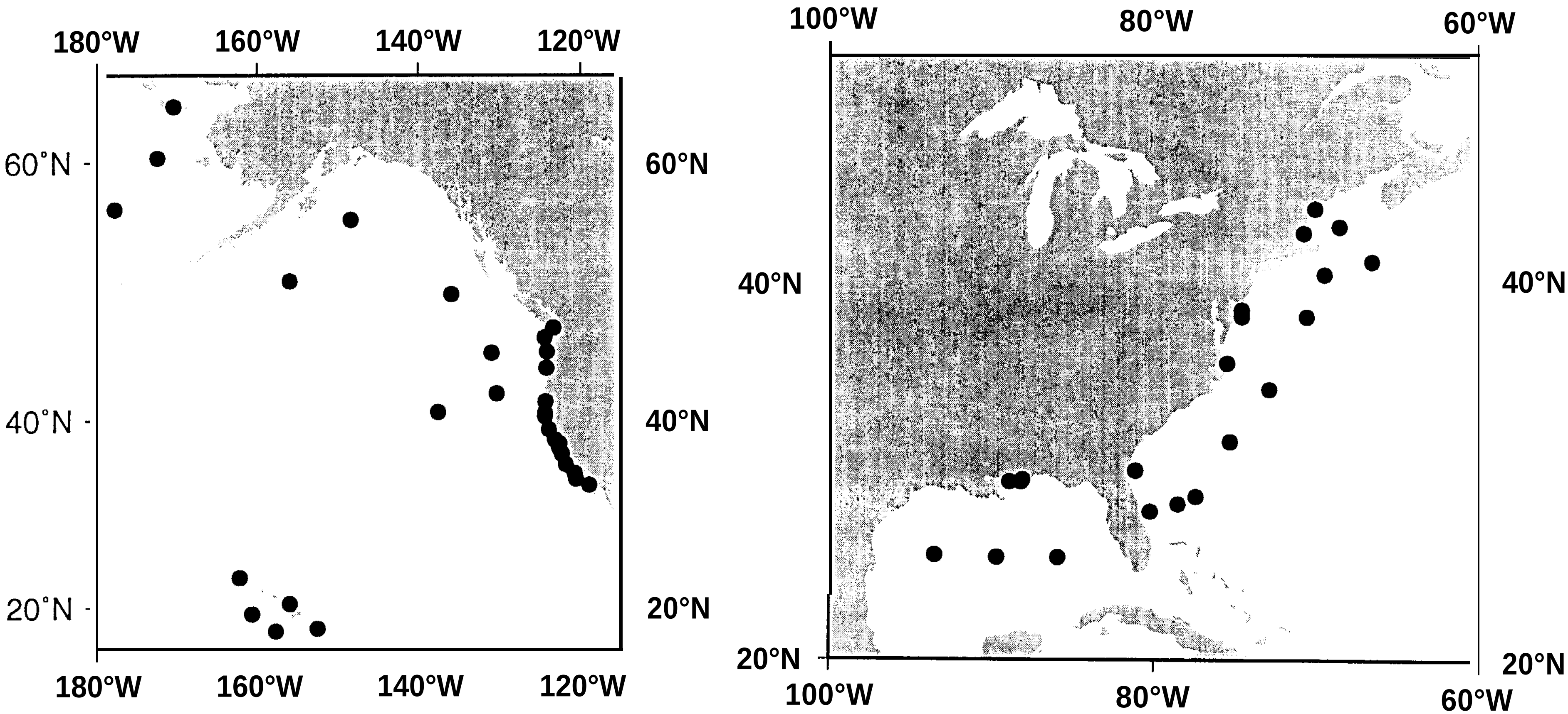
1. NSF Disks – There have been instances of NSF disks not being available when needed. This appears to be a problem at the system level rather than with implementation.

D.3 Matchup Database Problems

None listed separately

D.4 DSP And Headquarters Related Problems

None listed.



Western Buoys

Eastern Buoys

FIGURE B.3.1 NDBC MOORED BUOYS

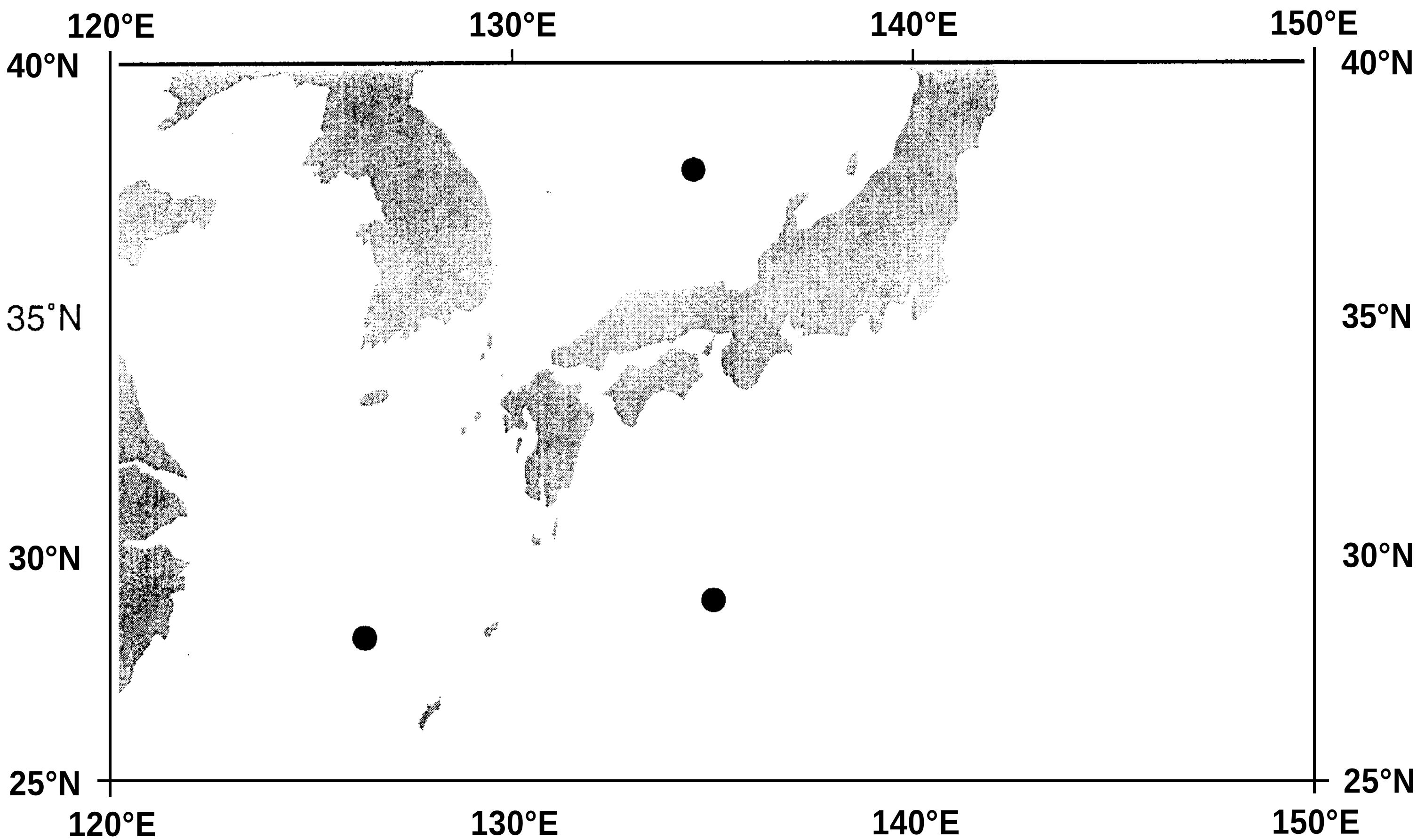


FIGURE B.3.2 JAPANESE METEOROLOGICAL AGENCY MOORED BUOYS

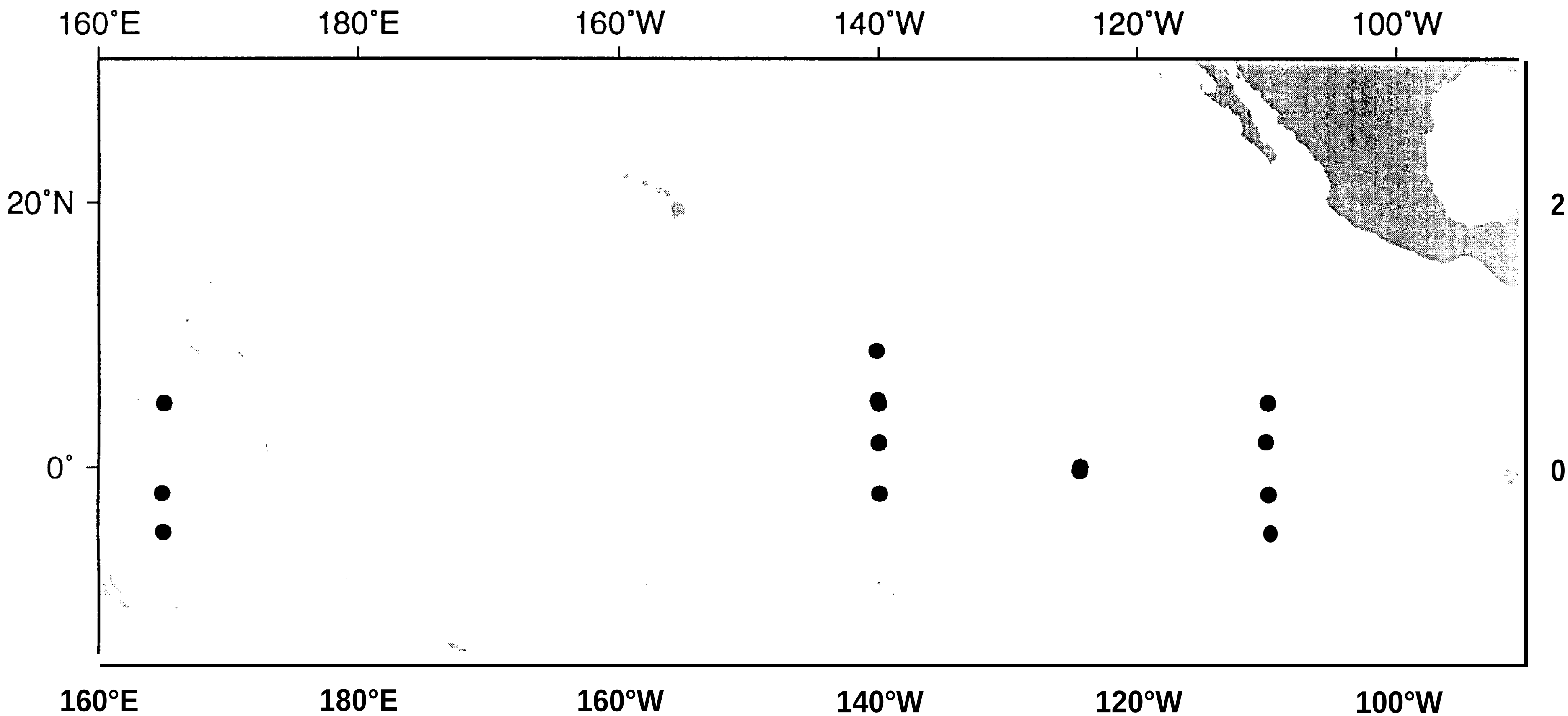


Figure B.3.3 TOGA / TAO MOORED BUOYS

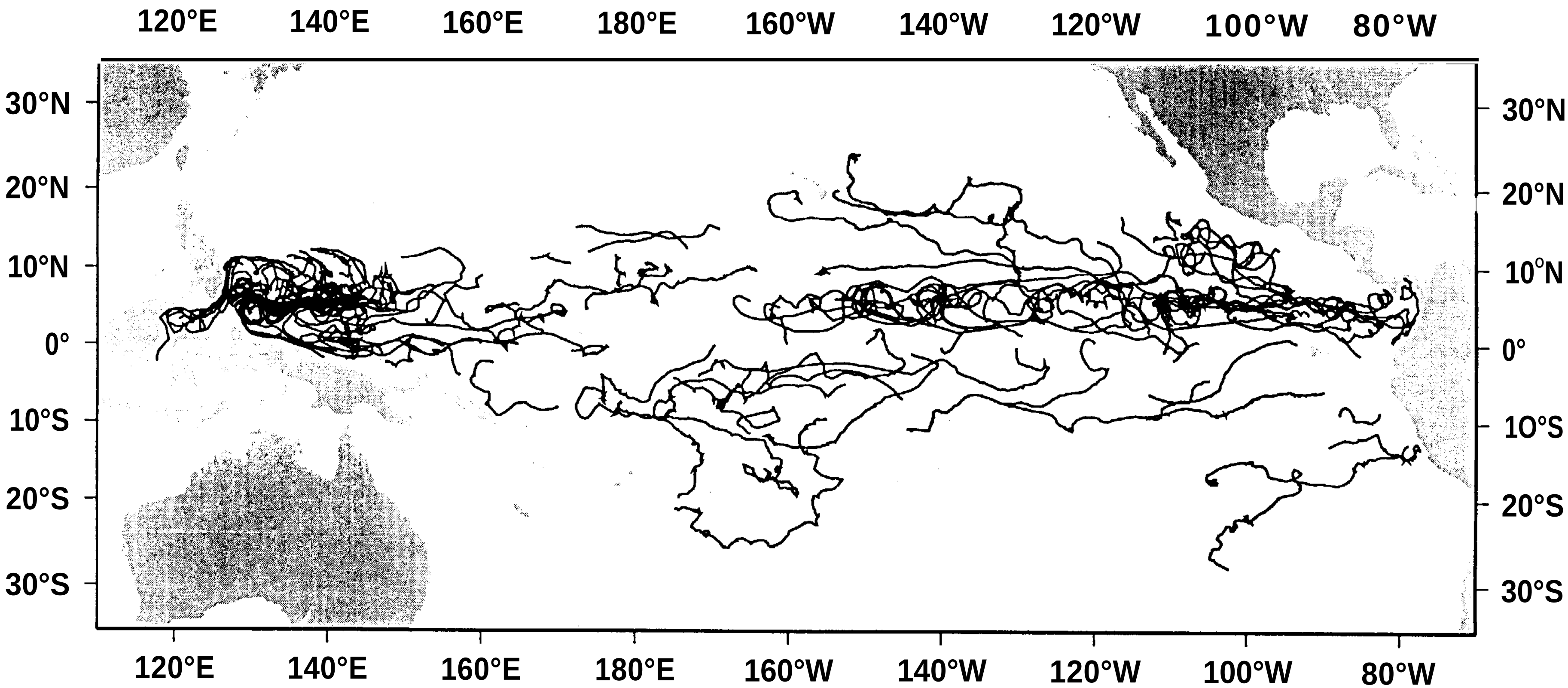


Figure B.3.4 NOAA/AOML DRIFTING BUOY TRACKS

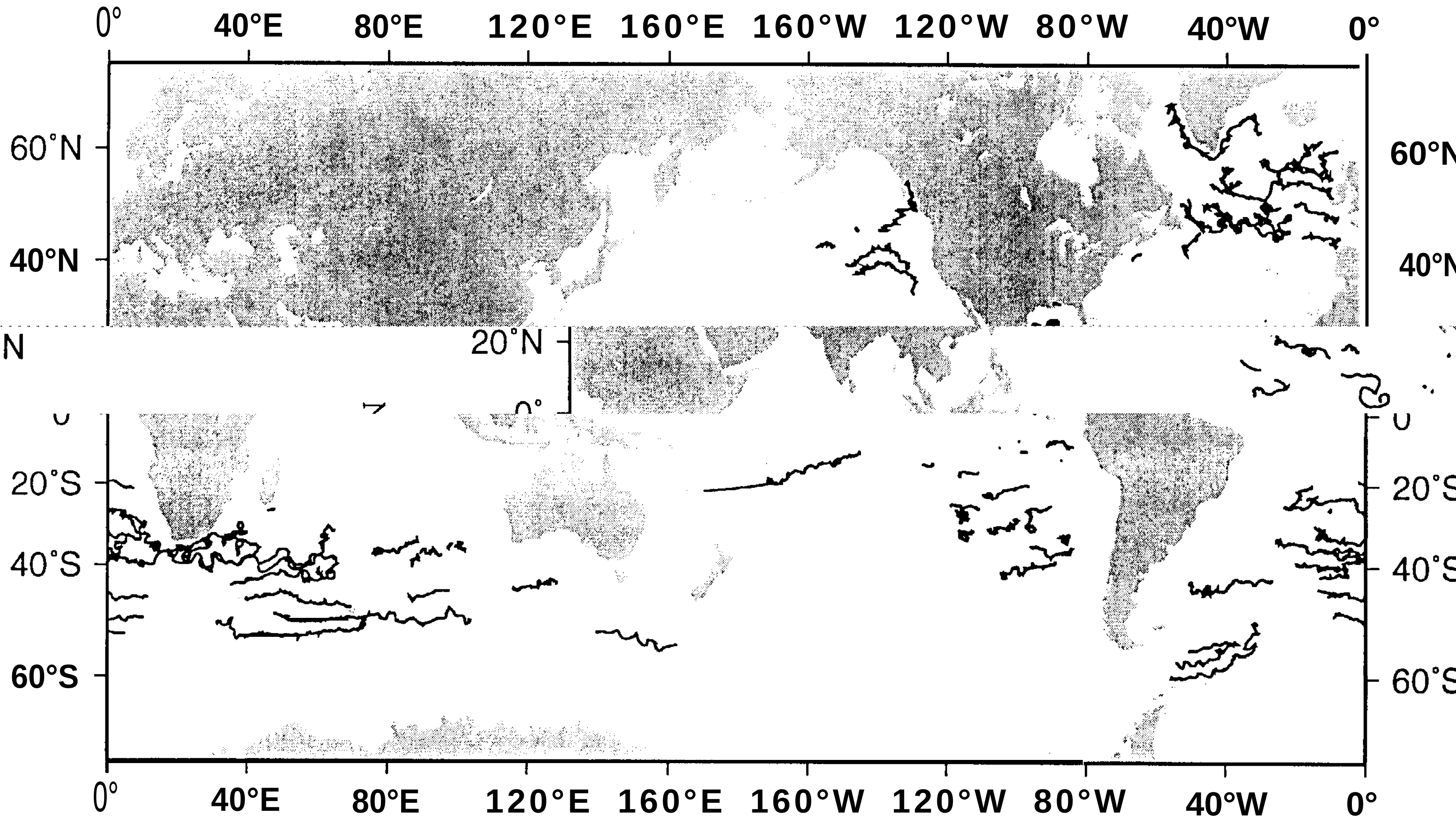


Figure B.3.5 MEDS DRIFTING BUOY TRACKS